Lab syllabus for SOC361: Statistics for Sociologists II

Fall 2024

Griffin Bur
gjmbteaching@gmail.com
Lab times: M 9:55 AM - 11:50 AM; M 1:20 PM - 3:15 PM
Lab locations: Sewell SSB 3218 (both)
Office hours: Th, 12PM-2PM, Sewell SSB, 2413 (book here)
Lab website: this Github repo

1 What we do here

We will generally review the course content in the orbit of the lecture (sometimes a bit before, sometimes a bit after). I anticipate that lab will have a fairly heavy does of Stata content, but we will also discuss the pure math. At any rate, every lab will cover the previous week's homework.

2 Course modality

This is a real-time, in-person laboratory. Come to lab—it will generally be fun, and showing up to things in person, prepared and in fine fettle, is also a matter of professionalization. Occasional absences are OK; you don't need to e-mail me. I do reserve the right to factor in lab attendance into liminal cases (e.g., if you're on a grade cusp, I will lean one way or the other based on this).

I generally cold-call people if there are not active volunteers. I learned early on in teaching that students actually generally prefer this. I'll use a random number generator to give me a permutation (ordered list without repeats) of names to call each lab. If you are yourself a lecturer or TA, as some of you are, I recommend giving it a go even if it seems counterintuitive.

3 Communication

All communication for the class should go to the email listed above, which is specific to the teaching of this course—please **do not e-mail** me at my Wisc.Edu address (UW's spam filter is disastrously bad).

4 Resources

4.1 This course

I generally will post data-sets, code, and any outlines of proofs to my Github (linked above). *Although my* ***lecture*** *materials consist in thorough, nicely LaTeX-formatted documents and thoroughly-annotated code, my lab notes are generally more minimal.* I'll typically do the pure math components on the board and post rough outlines, but I expect students to take active notes; similarly, rather than provide you with fully annotated do-files, I will generally post outlines with some minimal comments. I suggest either making notes in such a file for each week or having one running script.

4.2 Background knowledge

I've lectured SOC 360 many times and have also lectured 357 and 365 a couple times. If you want help with any of these prerequisites, all of my lecture notes, code, and taped lectures for those courses are available here.

5 Problem-solving, seeking help, and general workflow tips

In this section, I'll just mention a few general guidelines for how to solve problems on your own, how to seek help from me when you need to, and some overall guidance on workflow. ***These are skills which I learned later in graduate school and wish I had learned earlier. Learn from my mistakes!***

5.1 Problem-solving on your own

There is a strong convention for seeking advice on online math and computing fora. I'm going to summarize the basics of that etiquette below. Please follow it. I am *more than happy to answer questions*—indeed, if you read my public-facing teaching reviews, you'll see that many students think I am very generous—but at a higher level, I want to establish good norms about what to do *first*. These are computing-oriented but apply *mutatis mutandi* to math.

Here are just a few common Stata "gotchas". I could list many more, and some of these may not make sense to you yet, but consider looking at this list later. The first major gotcha is this: ***Did you Google it and look at a few Statalist posts? Did you see the SSCC website? Did you look at any of my example code or code from past semesters***?

If you did, and you didn't find an answer, did you make any of the following common goofs (don't worry; we all have done it)?

- Did you make a working directory and put data there?
- Did you remember line-extenders for multi-line commands? → `///`
- Did you misspell a command or variable? Did you forget a comma after the command and before the options? Did you start and finish parentheses where needed and close quotation marks?
- Did you look at the help file to check the syntax? (help command).

- Did you remember that variables are case-sensitive? `MPG != mpg`
- Did you remember that MVs are technically infinitely-large? Stata excludes them from most commands, but when you make a new variable, you must be very careful here.
- Did you remember that "`=`" assigns in most languages and it is actually "`==`" which evaluates?
- More fundamentally, are you even sure what it is that you're actually trying to do? Or is what you're trying to do inadvisable, impossible, or inefficient?

5.2 How to ask for help

When it comes time to ask, please do the following…

1. Attach your code in a do-file; comment on the part that doesn't work. *Do not send a screenshot*; this makes it much harder for me to actually run your code.
2. Attach or link to any data that you're using unless you know I have it because I gave it to you.
3. Describe in the e-mail what you wanted to happen, what *did* happen, and what you've tried to do to fix the problem. Don't worry about the error message if it's not very specific; Stata is generally very user-friendly, but the explanation of errors can be cryptic.
4. Allow me 24 hours to reply.

By the way, as a bit of motivation, why learn code?.

5.3 Workflow tips

5.3.1 File organization, computer usage

One of the ancillary benefits of this class is to teach people how to organize data, and information in general, efficiently. As computers have gotten more and more sophisticated, the barrier to entry for using them at a basic level has become practically zero.

Here's the problem: the skills you need to be a researcher or succeed in the white-collar workforce require more than that. Much like we do not teach basic grammar or the use of a slide-rule anymore, we do not teach basic file-system management. And, as in all of those other cases, while we *have* developed a social system where most people don't *need* those skills, you still need them in order to be a top-tier communicator, a competent practical user of math, or a skilled office worker—and the unfortunate fact is that you now need to learn those skills quickly, instead of over a lifetime.

I'll teach these skills as we go on our way in the lab, but if "file management" sounds like Greek to you, please watch this video by Jeff Su, which will immediately improve your life. I make many of the same points in my first two videos on Stata, which you can watch now (start here).

For this class, what do you concretely need to do to succeed? I'll show this on Day 1, but you should…

1. Have folder on your course for this class, with subfolders for data, do-files, homeworks, and your projects. Or, you can organize by week; either way, you probably need a subfolder system to keep track of it easily. I recommend locating this on your desktop.

2. Actually put the documents for our course in the correct place in that folder, not just in your downloads.
3. Always navigate to that folder when writing a do-file (see below). Although you *can* open data in Stata easily by just clicking on the .dta file (since no other program on your computer will open it), you should avoid this habit.
4. Name your files in a rational way. It's ideal for me and you if you label things like "YYYY-MM-DD-title-separated-by-dashes". Labelling files in YMD fashion makes them much easier to organize. Please at least label do-files that you send to me in this way.

5.3.2 Do-file etiquette

Your do-files should always have the standard header, and this is important enough that it might as well go in the syllabus for easy reference. The standard header is this:

```
* DATE: YYYY-MM-DD
* AUTHOR: $your name$ [dollar signs in code mean "user input goes here"]
* PURPOSE: $your purpose$
version $your version$
cd $"/path/to/your/folder"$
capture log close
log using $file name: generally give it the same name as the do-file$, text replace
* Always write comments as necessary. You can write them with an asterisk
* which must be rewritten each line and can't come after commands
// with two forward slashes, which /can/ come after commands
/* or this "parenthesis" type of comment, which runs until you say "stop" with a */
code goes here
* comment when needed
```

5.3.3 General Stata tips

1. Learn your hotkeys. Run do-files (at least on Mac) with CMD+SHIFT+D. Open a do-file from the command line with `doedit`. ***Crucially, if you just want to run one block of code, not the whole file, just carefully highlight that block.***
2. Don't rewrite commands that just need a tweak when working in the command window. Just click on that old command from the history tab, then change.
3. Write a loop if you don't want to type something tediously by hand.
4. Use the help files; they are much better than those for any other comparable software, and much more user-friendly.
5. ***Generally avoid saving changes to data; instead, put the changes that you want to make into a do-file, and then just run that do-file. This ensures that your work is reproducible if someone just has your code and a public data-set.***